CLAIMS

Having thus described our invention, what we claim as new and desire to secure by Letters Patent is as follows:

1. A method of improving the scalability of real-time collaboration among clients in a peer-to-peer network comprising the step of providing a timestamp and priority-based serialization protocol that can substitute for a centralized server-based serialization protocol of a real-time collaboration session.

2. The method of claim 1, wherein the timestamp used is based on one global clock which is distributed and kept synchronized among the clients participating in the collaboration session.

3. The method of claim 2, wherein clock distribution and maintenance of clock synchrony is done by Network Time Protocol (NTP).

4. The method of claim 2, wherein clock distribution and maintenance of clock synchrony is done by Simple Network Time Protocol (SNTP).

5. The method of claim 2, wherein clock distribution and maintenance of clock synchrony is done by an interactive convergence protocol.

6. The method of claim 2, wherein clock distribution and maintenance of clock synchrony is done by of manual intervention and manual cues.

7. The method of claim 1, wherein clients are fully connected to each other by first-in, first-out (FIFO) communication channels.

8. The method of claim 1, wherein incorrect serializations of modifications can occur, which then can be undone and corrected using a rollback mechanism.

9. The method of claim 8, wherein rollback of serialization decisions have well-defined and known, upper and lower time/timestamp bounds.

10. The method of claim 9, including optimizations which eliminate a need for rollback when an accompanying latency and communication costs are acceptable.

11. The method of claim 9, including checkpoints in order to provide additional safety and reduce memory requirements arising from the rollbacks.

12. The method of claim 9, wherein checkpoints can be all be locally stored by each client, or shared by multiple clients with say only one checkpoint storage for the multiple clients, the multiple clients sometimes being restricted to being only neighbors of each other.

13. The method of claim 1, wherein as long as there is at least one client present in a collaboration session at any time, any client participating in the collaboration session can be either dynamic or static, which means that either the client can participate in the collaboration session from start to finish, or it can join and/or leave the collaboration session while the session is ongoing.

14. The method of claim 13, wherein dynamic joining of clients is based on a checkpoints mechanism.

15. The method of claim 14, including an optimization wherein an introducer for a dynamically joining client provides a more developed version of a workspace than a checkpoint identified for the joining client, thereby reducing computation, space requirements and communication requirements for the joining purpose.

16. The method of claim 14, wherein a more developed version of a workspace provided by an introducer can comprise a checkpoint identified for joining, developed further by incorporating all serialized modifications available with the introducer up to or before a rollback window for the introducer at the time of communicating the workspace to the joining client.

17. The method of claim 1, wherein multiple definitions of a modification are supported, including partitioning-based modifications.

18. The method of claim 17, wherein partitioning-based modifications are fully supported, including inter-partition synchronisation via modifications over multiple partitions, wherein multiple partitions can comprise all kind of partition hierarchies and partition groups.

19. The method of claim 1, wherein locking and unlocking of workspace partitions are supported.

20. The method of claim 19, wherein the support for locking and unlocking reuses a serialization mechanism.

21. The method of claim 1, including an optimization for light-weight clients wherein a back-end process takes over storage intensive aspects of serialization that would ordinarily be carried out by the clients themselves.

22. The method of claim 1, including a method of dynamically switching to a distributed server and back in order to utilize a distributed server for periods of network response when a distributed server is better suited to supporting real-time collaboration than the serialization protocol.

23. The method of claim 1, wherein interoperability is improved across heterogeneous software/hardware platforms by improving efficiency and scalability of real-time collaboration without relying on any specialized support from the network/back-end supporting the real-time collaboration.

24. The method of claim 1, wherein interoperability in heterogeneous environments is improved by being able to work in conjunction with a distributed server for providing an improvement in the efficiency/scalability/ throughput of real-time collaboration.

25. The method of claim 1, wherein interoperability in heterogeneous environments is improved by including special support via optimizations and methods oriented towards lightweight clients suited to pervasive devices, which are likely to comprise a large part of heterogeneous environments in the near future.